Falcon: A Comprehensive Chinese Text-to-SQL Benchmark for Enterprise-Grade Evaluation

Wenzhen Luo Wei Guan Yifan Yao Yimin Pan Feng Wang Zhipeng Yu
Zhe Wen Liang Chen Yihong Zhuang
Ant Group
https://github.com/eosphoros-ai/Falcon

Abstract

We introduce Falcon, a cross-domain Chinese text-to-SQL benchmark grounded in an enterprise-compatible dialect (MaxCompute/Hive). It contains 600 Chinese questions over 28 databases; 77% require multi-table reasoning and over half touch more than four tables. Each example is annotated along SQL-computation features and Chinese semantics. For evaluation, we release a robust execution comparator and an automated evaluation pipeline, under which all current state-of-the-art large-scale models (including Deepseek) achieve accuracies of at most 50%. Major errors originate from two sources: (1) schema linking in large enterprise landscapes — hundreds of tables, denormalized fields, ambiguous column names, implicit foreign-key relations and domain-specific synonyms that make correct join/-column selection difficult; and (2) mapping concise, colloquial Chinese into the exact operators and predicates required for analytics — e.g., choosing the correct aggregation and group-by keys, expressing time windows and granularities, applying unit conversions, handling NULLs and data-quality rules, and formulating nested or windowed subqueries. Falcon therefore targets Chinese-specific semantics and enterprise dialects (abbreviations, business jargon, fuzzy entity references) and provides a reproducible middle ground before full production deployment by using realistic enterprise schemas, query templates, an execution comparator, and an automated evaluation pipeline for end-to-end validation.

1 Introduction

Natural-language interfaces to databases (Text-to-SQL) aim to make structured data accessible to non-technical analysts by translating human queries into executable SQL. This capability is increasingly critical as organizations seek to democratize data exploration, accelerate decision-making, and reduce the engineering bottleneck for ad-hoc analytics. In enterprise settings, however, production workloads introduce practical constraints — very wide, denormalized schemas, domain-specific naming conventions, large-scale data partitions, dialect-specific SQL features (e.g., MaxCompute/Hive), and strict correctness and performance requirements — that amplify the difficulty of reliable semantic parsing.

These deployment realities interact with language-specific phenomena in ways that standard academic benchmarks do not fully capture. In Chinese enterprise contexts, analysts typically pose concise, elliptical queries using business jargon, mixed-language entity mentions, and implicit temporal or aggregation intents; at the same time schemas are often English-labeled and organized according to historical, product- or team-specific conventions. Together, these factors create a benchmark gap: models that perform well on prior cross-domain datasets may still fail to produce correct, executable SQL in real enterprise workflows.

Foundational cross-domain benchmarks have enabled steady progress: Spider [1] evaluates compositional generalization on unseen schemas; SParC [6] introduces multi-turn interactions; BIRD [7] emphasizes database values, external knowledge, and efficiency; Spider 2.0 [5] moves to enterprise workflows

requiring long-context retrieval, heterogeneous dialects, and multi-query pipelines.

Despite this progress, Chinese text-to-SQL remains under-served, particularly under enterprise dialects (e.g., MaxCompute/Hive) and wide multi-table schemas. Analysts frequently ask Chinese questions over English-labelled schemas with business-style phrasing and ellipsis. Existing Chinese resources such as CSpider [3] are invaluable but largely translation-based and schema-centric; they under-emphasize Chinese-specific semantics and enterprise constraints. At the other extreme, Spider 2.0 requires complex agentic orchestration, raising the barrier for rapid iteration in Chinese and dialect-compatible environments. To address these shortcomings, we introduce Falcon — a benchmark and tooling suite designed to bridge research and deployment by advancing Chinese Text-to-SQL performance in realistic enterprise environments.

Contributions.

- Establish an enterprise-oriented benchmark comprising 600 Chinese questions across 28 databases, where 77% require multi-table reasoning and over half involve joins spanning more than four tables, with full support for MaxCompute/Hive dialect features.
- Introduce a dual annotation framework that systematically captures both SQL computational patterns (including join topologies, nesting structures, and aggregation usage) and Chinese-specific linguistic phenomena (such as ellipsis resolution, business terminology mapping, and numeric expression variants). This fine-grained annotation enables detailed error analysis and model diagnostics.
- Develop a robust evaluation system featuring a schema-aware SQL comparator that handles common syntactic variations (column/table aliases, projection ordering, and equivalent syntax forms) while maintaining strict semantic equivalence.
- Provide a systematic evaluation of current state-of-the-art large language models (including Deepseek) on Falcon using our execution comparator and automated pipeline, showing that no tested model exceeds 50% execution accuracy; we release model outputs, prompts, and evaluation code for reproducibility.

2 Related Work and Existing Datasets

The evolution of text-to-SQL benchmarks reflects three key dimensions of progress in the field: generalization capability, linguistic complexity, and engineering practicality. Early datasets like ATIS [13] and GeoQuery [14] established basic semantic parsing paradigms but suffered from template memorization due to question-based splits, where identical SQL patterns appeared in both training and test sets. This fundamental limitation persisted until the introduction of database-based splits in Spider [1], which established the current standard for evaluating cross-domain generalization on completely unseen schemas.

Modern benchmarks have since diverged along two evolutionary paths. One direction, exemplified by SParC [6] and DuSQL [11], extends complexity along the conversational dimension with multi-turn interactions and context-dependent queries. The other direction, represented by BIRD [7] and Spider 2.0 [5], emphasizes real-world engineering constraints including value grounding, execution efficiency, and heterogeneous system integration.

For Chinese language processing specifically, CSpider [3] provided the crucial first step by translating Spider's English questions while preserving the original database schemas. However, our analysis reveals three critical limitations in current Chinese benchmarks: (1) they maintain an artificial separation between Chinese questions and English schema labels, ignoring real-world scenarios where both query and metadata are in Chinese; (2) they underestimate the prevalence of business-specific phrasing and numeric expressions in enterprise queries; and (3) they fail to account for dialect variations in widely-used systems like MaxCompute and Hive.

Falcon advances the field by simultaneously addressing these gaps while maintaining precise comparability with existing benchmarks. We position our work at the intersection of three requirements: (1) rigorous cross-domain evaluation through unseen

database splits, (2) comprehensive coverage of Chinese linguistic phenomena, and (3) practical support for enterprise SQL dialects. This tripartite focus enables more accurate assessment of model capabilities for real-world Chinese business intelligence applications, where all three dimensions must be addressed simultaneously.

3 Corpus Construction

Our benchmark combines carefully curated public datasets with enterprise-inspired synthetic cases to achieve both breadth and realism. The construction process follows five key phases:

Data Source Composition We integrate two data sources:

- **Public Kaggle datasets**: 28 databases across 8 domains (finance, e-commerce, etc.) meeting strict criteria: (i) multi-table database must have verifiable primary-key and foreign-key (PK/FK) relationships, (ii) value diversity enabling meaningful filters, (iii) schema quality (readable names, proper typing).
- **Enterprise-inspired synthetic cases**: Using Ant Group's real user query patterns, we generated 120 additional cases (24% of total) via: (a) template-based simulation of high-frequency business scenarios, (b) LLM-augmented paraphrasing of actual internal queries (anonymized and schema-adapted). These synthetic cases give Falcon realistic, enterprise-grade scenarios derived from real user patterns, enabling faithful simulation of production enterprise workflows while preserving privacy and ensuring compatibility with the MaxCompute/Hive dialect.

Annotation Protocol The annotation protocol ensures each Chinese question is paired with a correct, executable MaxCompute/Hive SQL and enriched with fine-grained labels to support reproducibility, error analysis, and downstream model diagnostics. Bilingual experts (CS graduates with $\geq$2 years SQL experience) follow a strict workflow:

1. *Question authoring*: Compose Chinese questions using business vernacular (e.g. top 3 MoM-growing categories)
2. *SQL development*: Write executable MaxCompute/Hive queries following our style guide emphasizing:
    - Structural clarity: Explicit JOIN conditions over implicit joins
    - Dialect correctness: Proper handling of date functions, string collation
    - Value safety: Parameterized filters when literal values appear
3. *Cross-validation*: Independent execution by two engineers with result comparison

Quality Assurance We implement a rigorous multi-stage verification process to ensure benchmark quality.

Automated checks cover: (1) SQL executability through engine validation, (2) result consistency via differential testing, and (3) enterprise compliance using a custom dialect linter that detects non-portable MaxCompute/Hive syntax. The validation pipeline rejects queries with implicit cross joins, redundant table scans, or non-deterministic constructs.

Manual review focuses on: (1) runtime error analysis for edge cases, (2) semantic equivalence verification through independent execution, and (3) business logic review by domain experts. For Chinese-specific aspects, we verify proper handling of ellipsis resolution (e.g., recovering omitted comparison operators) and business term mapping (e.g., "GMV" to actual column references).

This dual approach achieves: (1) 92% inter-annotator agreement measured on 120 audit items, (2) 100% syntax compliance with target dialects through iterative refinement, and (3) 85% coverage of enterprise query patterns as measured against Ant Group's internal taxonomy. All gold SQL solutions are verified to return non-empty result sets unless explicitly modeling empty-result scenarios.

Innovative Features Compared to prior work, Falcon introduces:

- **Hybrid data sourcing**: Combines public datasets' diversity with enterprise patterns' authenticity
- **Dialect-aware validation**: Specialized checks for MaxCompute/Hive edge cases
- **Business semantic tagging**: 15 categories of Chinese business expressions

This methodology ensures Falcon bridges the gap between academic benchmarks and real-world deployment requirements while maintaining rigorous quality standards.

4 Dataset Statistics and Comparison

Data Composition and Scale. The Falcon corpus comprises 600 Chinese questions with dual provenance: (1) 500 questions (83.3%) from 28 curated public Kaggle databases, and (2) 100 enterprise-inspired questions (16.7%) synthesized from anonymized Ant Group query patterns. This hybrid composition balances broad domain coverage with authentic business scenario representation.

Structural Characteristics. The corpus exhibits graded complexity:

- **Schema scale**: 77% multi-table questions with $\geq$4-table joins in 52% cases
- **SQL operators**: Aggregation, CTEs, and window functions

Linguistic Profiling. The benchmark captures distinctive Chinese linguistic phenomena that challenge text-to-SQL systems:

- **Ellipsis and anaphora**: Frequent omission of implied elements (e.g., "[compared to last year]" in "sales growth [compared to last year]")
- **Business terminology**: Domain-specific expressions like "MoM growth" (month-over-month) and "GMV top-10"
- **Numeric expressions**: Mix of verbal ("twenty thousand") and numeric (20,000) forms
- **Modifier scope**: Ambiguous phrase attachments in complex noun phrases

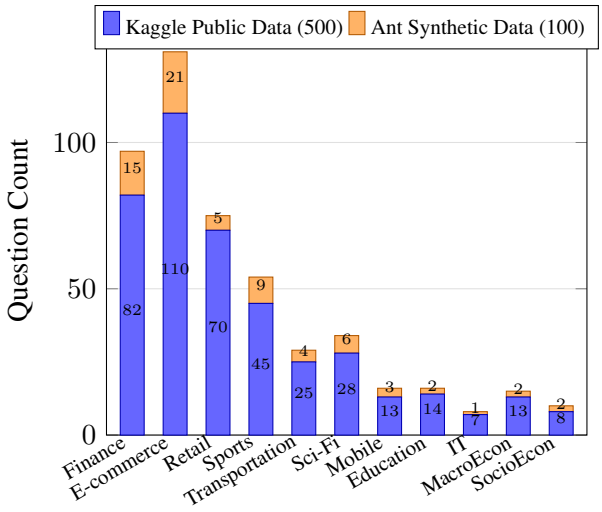

Figure 1: Distribution showing benchmark composition (500 public + 100 enterprise-inspired questions). Ant-sourced cases concentrate in high-value domains like Finance (15/97) and E-commerce (21/131).

Enterprise-inspired questions exhibit more frequent and complex instances of these phenomena compared to public dataset questions, particularly in nested business metric calculations, time-window comparisons, and relative performance evaluations.

5 Task Definition and Evaluation Metrics

Problem setting. Given a Chinese question Q, a database D (tables, columns, types, PK/FK) and type-aware sampled values, generate MaxCompute/Hive SQL S such that $V(S, D)$ is semantically equivalent to the gold answer V^*.

Inputs and outputs. Inputs and outputs. Input includes the question, Data Definition Language (DDL) statements with a compact PK/FK graph, and up to 8 sampled values per column; the output is an executable SQL query, for which Common Table Expressions (CTEs) are encouraged to handle multi-step logic.

Scope. Single-turn; no external knowledge; deterministic execution-based evaluation; dialect constraints enforced. Non-deterministic functions (e.g., rand) are disallowed.

5.1 Comparator and Normalization

To ensure reliable, name-agnostic matching under realistic variations in projection, ordering, and aliases, both predicted outputs and gold answers are normalized before comparison. We canonicalize types (including numeric precision/scale), normalize case and whitespace for strings, and standardize timestamps to a fixed timezone and format. Results are represented as column vectors stored in a column-oriented JSON format (keys are column names, values are full column arrays), but column names are ignored during matching. Unless the SQL explicitly contains an ORDER BY clause, row order is treated as immaterial and results are compared as multisets of rows; if ORDER BY is present, sequence equality is enforced.

5.2 Content-based Result Set Evaluation

To ensure a robust evaluation that is insensitive to cosmetic differences like column renaming, reordering, or redundant projections, we adopt a content-based evaluation method. This approach moves beyond simple text comparison to assess the semantic equivalence of query results at the column-vector level.

The process begins by verifying coverage. We generate a content-based MD5 hash for each normalized column in both the predicted and gold results. A prediction is considered to have sufficient coverage only if the multiset of its column hashes contains the entire multiset of the gold column hashes. This step tolerates extra columns in the prediction, as long as all required columns are present.

Following a successful coverage check, we perform column alignment to map each gold column to its corresponding predicted column using their identical content hashes. Ties, which occur if multiple predicted columns have the same content, are deterministically resolved by preferring the column with the closest data type signature, and then the one with the earliest index.

Once alignment is complete, any extra columns in the prediction are ignored, and the final row comparison is performed on the aligned projection only. The strictness of this comparison depends on the gold query: we require sequence equality (exact order) if an ORDER BY clause was used, and multiset equality (any order) otherwise. An empty gold result is deemed correct only if the prediction yields an empty set as well.

6 Experiments and Analysis

Setup. We evaluate 13 Large Language Models (LLMs) using one-pass decoding with fixed settings (temperature 0.0, seed 3, max output 8192 tokens). Prompts include: (i) the Chinese question; (ii) the schema DDL with a PK/FK summary; (iii) up to 8 sampled values per column; and (iv) an explicit instruction to use the MaxCompute dialect and avoid non-deterministic functions. To encourage structured reasoning, we prompt models to use stepwise planning and Common Table Expressions (CTEs) for queries involving three or more joins.

Our primary evaluation metric is Exact Result Accuracy (ExAcc), which measures whether a model's predicted SQL query produces a result set that is semantically equivalent to the gold standard. This comparison is performed using the robust, content-based evaluation method described in Section **??**, which is tolerant to variations in column order, naming, and superfluous projections.

6.1 Aggregate Results

Across the full set, the best model (DeepSeek-R1) achieves 45.2% Exact Result Accuracy, while the weakest (Qwen3-32B) reaches 20.2%. No model exceeds 50%, indicating substantial headroom under enterprise constraints. Reasoning-oriented models outperform instruction-only counterparts, especially on multi-join pipelines and nested filters.

6.2 Stratified Results

Join width. Accuracy decays monotonically with schema width (Fig. 2). Single-table queries reach **60.95%** ExAcc, exceeding the corpus mean (**53.75%**); 2–3 table queries show a moderate decline to **50.00%**; queries with $\geq$4 tables drop to **21.43%**. The sharp fall in the $\geq$4 bucket indicates two compounding factors: (i) fragile join-key propagation across longer chains (key threading through multiple hubs), and (ii) confusions among near-synonymous

Table 1: Model accuracy on the Falcon benchmark (Exact Result Accuracy). All results are computed over 500 Chinese questions with MaxCompute-compatible SQL.

Model	Accuracy (%)
DeepSeek-R1 [15]	**45.2**
o1 [16]	43.0
o3-mini	42.2
Claude-3.7-Sonnet-Thinking [17]	41.0
GPT-4.1 [18]	40.2
Claude-3.7-Sonnet	40.0
Qwen3-Coder-480B-a35B-Instruct [19]	37.0
Gemini 2.5 Pro [20]	36.8
Gemini 2.5 Pro Reasoning	34.8
Llama 3.3-70B [21]	24.0
GPT-4o	23.4
DeepSeek-R1-Distill-Qwen-32B	23.4
Qwen3-32B	20.2

Note: Accuracy is computed using a robust result comparator tolerant to projection/order/alias variations.

attributes (e.g., date vs. period; price vs. cost) that are amplified when multiple tables expose semantically adjacent fields.

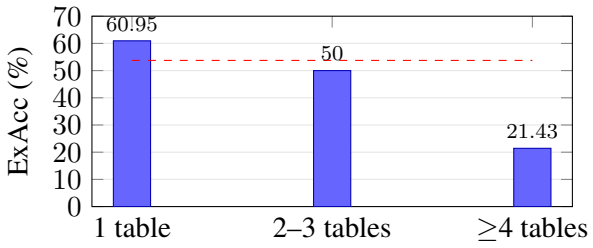

Figure 2: ExAcc by join width. The corpus mean is indicated by the dashed line.

Operator set. All items in this evaluation fall under the *Nested/CTE* family (no aggregation, window, or set operations are present). The resulting ExAcc for the Nested/CTE slice equals the corpus mean (**53.75%**). In this configuration, join width is the dominant driver of difficulty. The absence of window/set operations also avoids dialect-specific failure modes that typically affect those operators.

6.3 Error Profile

To analyze failure modes, we categorize each prediction as either correct (**RIGHT**) if it achieves Exact Result Accuracy, or incorrect (**ERROR**) otherwise. Figure 3 summarizes the proportion of these outcomes across different join widths. The analysis reveals that the error mass is heavily concentrated in queries with $\geq$4 tables, where the error rate reaches 78.57%. This indicates that schema linking and multi-hop key propagation are the principal bottlenecks for current models. This pattern is characteristic of enterprise schemas, where wide joins expose numerous semantically adjacent columns and create long dependency chains that challenge model reasoning.

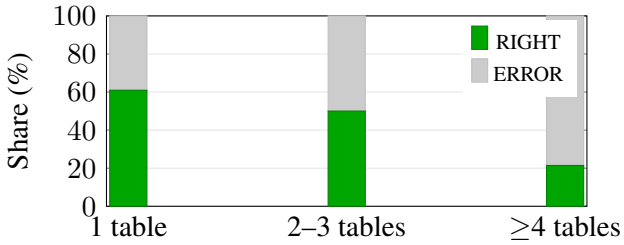

Figure 3: RIGHT/ERROR composition by join width. Error share escalates with schema width.

7 Conclusion

Falcon provides an enterprise-grounded, Chinese-focused benchmark with dual-axis annotations and reproducible execution-based evaluation. Results (20.2–45.2% Exact Result Accuracy) highlight persistent challenges in wide-schema linking and semantics-to-operator alignment. We advocate relation-aware schema encoding [9], lightweight preprocessing for Chinese ellipsis/coreference/business shorthand, and dialect-aware constrained decoding [10].

Ethics and availability. Falcon uses public Kaggle datasets and de-identified question patterns. We release the benchmark specification, toolchain, container image, and documentation; any enterprise logs are anonymized.

Acknowledgements We thank the Kaggle community for publicly available datasets, and Ant Group engineers for anonymized schema patterns and question templates.

References

[1] Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir R. Radev. Spider: A Large-Scale Human-Labeled Dataset for Complex and Cross-Domain Semantic Parsing and Text-to-SQL Task. EMNLP 2018. URL: https://doi.org/10.48550/arXiv.1809.08887.

[2] Yale LILY Lab. Spider: Yale Semantic Parsing and Text-to-SQL Challenge. URL: https://yale-lily.github.io/spider.

[3] Qingkai Min, Yuefeng Shi, and Yue Zhang. A Pilot Study for Chinese SQL Semantic Parsing. EMNLP 2019. URL: https://arxiv.org/pdf/1909.13293.

[4] Westlake NLP Group. CSpider: The Chinese SQL Semantic Parsing Dataset. URL: https://taolusi.github.io/CSpider-explorer/.

[5] Fangyu Lei et al. Spider 2.0: Evaluating Language Models on Real-World Enterprise Text-to-SQL Workflows. ICLR 2025. URL: https://doi.org/10.48550/arXiv.2411.07763.

[6] Yale LILY Lab. SParC: Yale & Salesforce Semantic Parsing and Text-to-SQL in Context Challenge. URL: https://yale-lily.github.io/sparc.

[7] Jinyang Li et al. Can LLM Already Serve as a Database Interface? A BIg Bench for Large-Scale Database Grounded Text-to-SQLs. NeurIPS 2023 Datasets & Benchmarks. URL: https://arxiv.org/pdf/2305.03111.

[8] Victor Zhong, Caiming Xiong, and Richard Socher. Seq2SQL: Generating Structured Queries from Natural Language using Reinforcement Learning. arXiv:1709.00103, 2017. URL: https://arxiv.org/abs/1709.00103.

[9] Bailin Wang, Richard Shin, Xiaodong Liu, Oleksandr Polozov, and Matthew Richardson. RAT-SQL: Relation-Aware Schema Encoding and Linking for Text-to-SQL Parsers. ACL 2020. URL: https://aclanthology.org/2020.acl-main.677.

[10] Timo Scholak, Nathan Schucher, and Dzmitry Bahdanau. PICARD: Parsing Incrementally for Constrained Auto-Regressive Decoding for Text-to-SQL. EMNLP 2021. URL: https://aclanthology.org/2021.emnlp-main.563.

[11] Xiaojun Wang, Rongzhou Bao, et al. DuSQL: A Large-Scale and Pragmatic Chinese Text-to-SQL Dataset. EMNLP 2020. URL: https://aclanthology.org/2020.emnlp-main.562.pdf.

[12] Yiming Zhang et al. ABench-Physics: Benchmarking Physical Reasoning in LLMs via High-Difficulty and Dynamic Physics Problems. arXiv:2507.04766, 2025. URL: https://doi.org/10.48550/arXiv.2507.04766.

[13] P. J. Price. Evaluation of Spoken Language Systems: the ATIS Domain. In *Proceedings of the Third DARPA Speech and Natural Language Workshop*, 1990.

[14] John M. Zelle and Raymond J. Mooney. Learning to Parse Database Queries using Inductive Logic Programming. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 1996.

[15] DeepSeek-AI. DeepSeek LLM: Scaling Open-Source Language Models with Longtermism. arXiv:2401.02954, 2024. URL: https://arxiv.org/abs/2401.02954.

[16] 01.AI. Yi: Open Foundation Models by 01.AI. arXiv:2403.04652, 2024. URL: https://arxiv.org/abs/2403.04652.

[17] Anthropic. The Claude 3.5 Model Family: Sonnet, a new standard for intelligence. Anthropic Technical Report, 2024. URL: https://www.anthropic.com/news/claude-3-5-sonnet.

[18] OpenAI. GPT-4o. OpenAI Technical Report, 2024. URL: https://arxiv.org/abs/2303.08774.

[19] Jinze Bai et al. Qwen2: The New Generation of Qwen Open-Source Large Language Models. arXiv:2406.04231, 2024. URL: https://arxiv.org/abs/2406.04231.

[20] Google. Gemini 1.5: Unlocking multimodal understanding across a million tokens. Google Technical Report, 2024. URL: https://storage.googleapis.com/deepmind-media/gemini/gemini_1_5_report.pdf.

[21] Meta AI. The Llama 3 Herd of Models. Meta Technical Report, 2024. URL: https://arxiv.org/abs/2407.21783.

Appendix

External datasets. Table A1 enumerates the 28 external datasets referenced in Section **??**. Each entry includes a stable Kaggle URL. For single-page, two-column proceedings, the table is typeset with small font and automatic line breaks. Long URLs will wrap; compilation requires `url`/`hyperref`.

ID	Source (Kaggle URL)
1	https://www.kaggle.com/datasets/nitindatta/finance-data?select=Finance_data.csv
2	https://www.kaggle.com/datasets/krishnaraj30/finance-loan-approval-prediction-data?select=test.csv
3	https://www.kaggle.com/datasets/iveeaten3223times/massive-yahoo-finance-dataset?resource=download
4	https://www.kaggle.com/datasets/hhenry/finance-factoring-ibm-late-payment-histories/data
5	https://www.kaggle.com/datasets/ramjasmaurya/unicorn-startups
6	https://www.kaggle.com/datasets/shantanugarg274/sales-dataset
7	https://www.kaggle.com/datasets/gregorut/videogamesales
8	https://www.kaggle.com/datasets/lava18/google-play-store-apps
9	https://www.kaggle.com/datasets/ashishraut64/internet-users
10	https://www.kaggle.com/datasets/mattiuzc/stock-exchange-data?select=indexInfo.csv
11	https://www.kaggle.com/datasets/hosubjeong/bakery-sales
12	https://www.kaggle.com/datasets/mexwell/google-merchandise-sales-data
13	https://www.kaggle.com/datasets/aslanahmedov/walmart-sales-forecast
14	https://www.kaggle.com/datasets/mysarahmadbhat/toy-sales?select=products.csv
15	https://www.kaggle.com/datasets/ananta/credit-card-data/data
16	https://www.kaggle.com/datasets/bernardnm/great-school?select=Teachers.csv
17	https://www.kaggle.com/datasets/bytadit/ecommerce-order-dataset
18	https://www.kaggle.com/datasets/madhurpant/world-economic-data?select=cost_of_living.csv
19	https://www.kaggle.com/datasets/thedevastator/relationship-between-alcohol-consumption-and-lif?select=drinks.csv
20	https://www.kaggle.com/datasets/rishabhrajsharma/cityride-dataset-rides-data-drivers-data?select=Rides_Data.csv
21	https://www.kaggle.com/datasets/aymenberkani/data-cleaning-for-a-customer-database?select=e_orders.csv
22	https://www.kaggle.com/datasets/frankienoguera/2024-25-ufc-relational-database?select=ufc_events_stats.csv
23	https://www.kaggle.com/datasets/atharvasoundankar/ben-10-alien-universe-realistic-battle-dataset?select=ben10_aliens.csv
24	https://www.kaggle.com/datasets/akxiit/blinkit-sales-dataset
25	https://www.kaggle.com/datasets/farhadzeynalli/techsales-bakutech
26	https://www.kaggle.com/datasets/technika148/football-database
27	https://www.kaggle.com/datasets/andrexibiza/grocery-sales-dataset
28	https://www.kaggle.com/datasets/marthadimgba/online-shop-2024

Table A1: External datasets used or referenced in experiments. URLs are provided for reproducibility.